CS xyz3-001 Foundations of Programming and Data Structures

Instructor Dr. Turgay Korkmaz

Homework 02

Due date: check BB

!!!! NO LATE HOMEWORK WILL BE ACCEPTED !!!

Write a C program that can display on what day a specific date (e.g., June 6, 6666) falls and display a twelve-month calendar for an arbitrary year. Specifically, your program should first prompt the user for an option

1: Find on what day a specific date falls.

2: Display the twelve-month calendar for a given year.

Option: (user enters choice here)

Depending on the given option, the program asks user to enter valid `mm dd yyyy` or just `yyyy` and call the necessary functions to print out the day or the calendar month by month. In either case the year `yyyy` should be greater than 1751 (Gregorian Calendar) (Find out why). You can use Perpetual calendar formula to determine the day for a specific date. For more details and example about this formula, see http://en.wikipedia.org/wiki/Perpetual_calendar You can use any other formula that you may find on the web too.

For example: if user selects option 1 and

enters: `2 15 2006`

your program should print

`Wednesday`

if user selects option 2 and

enters: `2009`

your program should print

```
  ***    CALENDAR for 2009   ***
January 2009
Sun  Mon  Tue  Wed  Thu  Fri  Sat
                        1    2    3
  4    5    6    7    8    9   10
 11   12   13   14   15   16   17
 18   19   20   21   22   23   24
 25   26   27   28   29   30   31

February 2009
Sun  Mon  Tue  Wed  Thu  Fri  Sat
  1    2    3    4    5    6    7
  8    9   10   11   12   13   14
                   .
                   .
                   .
```

Implementation Notes

Since we have not yet studied arrays, strings, or arrays of strings, you should design your algorithm to use **`if-else`** or **`switch`** statements to print the month or day names. You must partition your program into multiple functions. Here is an **example** partition (if you want you can modify it):–

- The function **`main()`** prompts the user for an option, depending on the option it asks a specific date mm dd yyyy or just year yyyy. Verify that input values are valid and year yyyy is greater than 1751. It is not enough to simply check 1<= mm <=12 and 1<= dd <=31. Depending on the month (e.g., February) dd should be less than 29 or 28 if the year is a leap year etc. If date is not valid, ask user to enter valid data.
- In case of option 1, the main function calls a function **`day = day_of_date(mm, dd, yyyy)`** that returns the day on which the given date falls. Using the returned value as a parameter, the main function calls another function **`printDayName( day )`** to print the name of the day.
- In case of option 2, the main function calls **`day = day_of_date(1, 1, yyyy)`** to determine the starting day of the year yyyy and then it invokes the function **`printCalendar(yyyy, day)`** to actually print the twelve month calendar.
- The function **`printCalendar()`** takes two arguments, the year number and the starting day. It then loops through the year and calls the function **`printMonth()`** twelve times, once for each month.
- The function **`printMonth()`** takes three arguments, the year number, the month number and the starting day of that particular month, and it returns the number of the day on which the next month starts. Print month has to first call a function **`printMonthName()`** and then print out the days of the month in calendar format.

  The function **`printMonthName()`** takes the year number and the month number as arguments, prints out the line identifying the month, and returns the number of days in that month. The example output of **`printMonthName()`** should look as above. Since we are not using arrays of strings, **`printMonthName()or printDayName()`** should use a switch statement to select and print the name of the month or day and to determine the number of days

in that month. If the month is February, it should also figure out whether the year is a leap year and return the correct number of days.

What to do and return: !!!! NO LATE HOMEWORK WILL BE ACCEPTED !!!

1. Create a directory `abc123-hw02`, using your own abc123. Do all your work under that directory.

2. Follow the problem-solving methodology to solve the problem(s). Then convert your solution(s) to a C program. You can name your program here as `hw02.c`

```
/*
 * Don't forget to include comments about the
 * problem, yourself and each major step in your
 * program! so that we can understand your
 * solution(s).
 */
```

3. Compile and run your program. Copy/paste the results in an output file, which you can name as `hw02-out.txt`

   To check your program correctness, compare your twelve month calendar with the one generated by cal yyyy program on Linux.

4. Zip the whole directory `abc123-hw02 as abc123-hw02.zip`
5. Go to BB Learn (http://learn.utsa.edu/) , login using your abc123
6. Submit your `abc123-hw02.zip` for hw02 under Assignments

You must submit your work using Blackboard Learn and respect the following rules:

1) All assignments must be submitted as either a zip or tar archive file unless it is a single pdf file.
2) Assignments must include all source code.
3) Assignments must include an output.txt file which demonstrates the final test output run by the student.
4) If your assignment does not run/compile, the output.txt file should include an explanation of what was accomplished, what the error message was that prevented the student from finishing the assignment and what the student BELIEVES to be the underlying cause of the error.
